



Revista Politécnica

ISSN: 1390-0129

editor.rp@epn.edu.ec

Escuela Politécnica Nacional

Ecuador

Simbaña, Wilson; Caiza, Julio; Chávez, Danilo; López, Gabriel
Diseño e Implementación de un Sistema de Monitoreo Remoto del Motor de un Vehículo
basado en Obd-II y la plataforma Arduino
Revista Politécnica, vol. 37, núm. 1, septiembre-marzo, 2016
Escuela Politécnica Nacional

Disponible en: <https://www.redalyc.org/articulo.oa?id=688773645004>

- Cómo citar el artículo
- Número completo
- Más información del artículo
- Página de la revista en redalyc.org

redalyc.org

Sistema de Información Científica

Red de Revistas Científicas de América Latina, el Caribe, España y Portugal

Proyecto académico sin fines de lucro, desarrollado bajo la iniciativa de acceso abierto

Diseño e Implementación de un Sistema de Monitoreo Remoto del Motor de un Vehículo basado en Obd-II y la plataforma Arduino

Simbaña Wilson¹; Caiza Julio¹; Chávez Danilo¹; López Gabriel¹

¹Escuela Politécnica Nacional, Facultad de Ingeniería Eléctrica y Electrónica, Quito, Ecuador

Resumen: En el presente proyecto se propone un sistema basado en OBD-II (On Board Diagnostics II System) que permite realizar el envío de información a un servidor remoto. La información de los transductores se obtiene a través del lector OBD ELM327, recolectándose los datos con un Arduino Mega 2560. Este tiene la capacidad de leer la información a través del uso de la librería obd.h. Luego, éste envía la información a un servidor remoto en Internet usando la red GPRS. El subsistema de recepción está configurado en un servidor remoto e incluye: un servicio para lectura y decodificación de la información recibida, una base de datos, y una interfaz web para gestión de la información que se encuentra almacenada. Los resultados muestran una operación efectiva de la solución, y cómo consecuencia todo el set de tecnologías podría ser utilizado para trabajar en nuevos trabajos relacionados.

Palabras clave: OBD-II, Arduino, ELM327, servidor remoto, GPRS network.

Design and Implementation of a Remote Monitoring System of a vehicle's engine based on Obd-II and the Arduino platform

Abstract: In this Project, an OBD-II (On Board Diagnostics II System) based system is proposed to transmit information to a remote server. The transmission subsystem obtains the data from sensors through the use of the OBD ELM327 reader, and it is processed using an Arduino Mega 2560. It is able to read the data by using the obd.h library. Then, it sends that data to a remote server on Internet, the transmission uses the GPRS network. The reception subsystem is set in a remote server and includes: a service to read and decode the received data, a database, and a web interface to check the stored information. The results showed an effective operation of the solution, and therefore, the whole set of technologies could be used to work in new related projects.

Keywords: OBD-II, Arduino, ELM327, remote Server, GPRS network.

1. INTRODUCCIÓN

En la actualidad, la industria automotriz impulsa nuevas tecnologías y soluciones prácticas con el propósito de mejorar el rendimiento, comodidad y la seguridad en los automóviles. Entre las innovaciones más importantes se encuentra la inclusión de los sistemas OBD-II, inicialmente orientado a monitorear los componentes del automóvil, y que ahora permite verificar que el funcionamiento y rendimiento del automóvil sea el óptimo.

A pesar de existir trabajos que se basan en OBD-II, se debe tener en cuenta que éstos utilizan una comunicación inalámbrica de corto alcance, convirtiéndose en una gran desventaja para el usuario ya que la única forma de poder monitorear los transductores del motor del automóvil es estando dentro del mismo.

Para poder tener la facilidad de que se pueda acceder al conjunto de datos desde cualquier lugar, se hace necesario el almacenar la información en un servidor remoto, y accesible

desde Internet. Una vez estén almacenados los datos, podrían ser procesados y brindar varios servicios futuros.

En este proyecto se propone el diseño e implementación de una solución que utiliza el microcontrolador para lectura de códigos OBD-II ELM327 para recolectar un conjunto de datos de la Unidad de Control del Motor (ECU, Engine Control Unit); la recolección de los datos y envío hacia el servidor remoto se realiza a través del dispositivo Arduino Mega 2560 con su respectivo módulo GPRS; el almacenamiento de los datos en el Sistema Gestor de Base de Datos (SGBD) MySQL; y finalmente la presentación de la información a través de una aplicación web desarrollada mediante PHP.

El artículo se compone de 5 secciones, la primera parte presenta las características principales, la descripción de componentes, los modos de medición, y los protocolos de comunicación del sistema OBD-II; la segunda parte presenta el diseño e implementación de la solución propuesta; la tercera parte detalla las pruebas de funcionamiento y resultados obtenidos de la solución propuesta, y la última presenta las conclusiones obtenidas.

gabriel.lopez@epn.edu.ec

2. TRABAJOS RELACIONADOS

La investigación de Cabala y Gamec (2012) aporta con la implementación de una conexión Bluetooth inalámbrica entre el dispositivo ELM327 que se conecta a una interfaz OBD-II y un dispositivo móvil, mismo que administra la obtención y recepción de datos del auto. Una vez que los datos han sido recibidos, el usuario los puede visualizar en diferentes presentaciones en el dispositivo móvil; dónde su objetivo primordial es dar a conocer el estado actual del vehículo. El presente proyecto se diferencia en que éste utiliza la tecnología ZigBee para conexión con el dispositivo ELM327 y realiza un envío de datos a un servidor remoto utilizando un Arduino Mega 2560; resaltando como objetivo principal el recibir información del vehículo para eventos de mantenimiento preventivo.

La investigación de Niazi et al. (2013), provee una solución para obtención de valores ODB relacionadas al estado del auto en tiempo real como: velocidad, revoluciones por minuto del motor, voltaje de la batería, etc. Además este sistema provee los códigos de problemas de diagnóstico (Diagnostic Trouble Codes - DTCs), mismos que tienen como objetivo verificar un posible mal funcionamiento del auto. El sistema utiliza un dispositivo basado en ELM327 para conectarse con la interfaz OBD-II, y este dispositivo se conecta a un computador por la interfaz USB, el cual posee un software especializado para desplegar los parámetros del auto. El presente proyecto se diferencia en que éste no depende de que un computador sea conectado directamente en el auto para medir su estado, ya que los datos son enviados a un servidor remoto mediante dispositivos integrados en el auto. Además, el presente proyecto realiza la obtención de datos orientado a la realización de un mantenimiento preventivo del auto, característica que el proyecto de Niazi et al (2013) no posee, dado que es un sistema reactivo a fallas del vehículo.

La investigación referente al Modelo de Predicción Inteligente del consumo de combustible del vehículo (Lee y Park, 2014) obtiene información por la interfaz OBD-II del estado del combustible utilizando tecnología basada en el dispositivo ELM327. Los datos obtenidos son procesados en base a un algoritmo utilizando un modelo de redes neuronales, que finalmente en un computador conectado mediante un puerto USB procesa y muestra la información de predicción. El proyecto propuesto se diferencia en que se enfoca en el envío de información a un servidor remoto y se toman varios valores del estado del auto, no sólo de combustible.

La investigación de Baghli et.al. (2012) toma datos del vehículo relacionados con el consumo de energía y corriente eléctrica instantánea requerida durante el trayecto de conducción utilizando la interfaz OBD-II. El estudio se realiza en autos híbridos, eléctricos y convencionales. La tecnología utilizada para la conexión con la interfaz OBD-II es ELM327 y ésta a su vez se comunica vía Bluetooth con un dispositivo móvil. El dispositivo móvil obtiene y almacena la información del vehículo/posición y a su vez ésta información es enviada a un servidor remoto. Finalmente esta

investigación en una segunda publicación (Baghli, Benmansour, y Djemai, 2014) evidencia que la información enviada a un servidor remoto a través del celular analiza los datos del auto en conjunto con la información de GPS del celular para mostrar el comportamiento del motor y consumo de energía. La diferencia con el proyecto propuesto son los dispositivos independientes instalados en el vehículo para el envío de información y no la dependencia de un dispositivo móvil conectado vía Bluetooth con el dispositivo ELM327 dentro del vehículo.

3. SISTEMA OBD-II

El sistema OBD-II se comenzó a utilizar de forma obligatoria por los nuevos automóviles en los Estados Unidos desde 1996, teniendo como objetivo monitorear los componentes que afecten el sistema de control de emisiones de gases contaminantes y medir parámetros en tiempo real como: temperaturas, presiones, velocidades entre otros (Meseguer, 2013).

Cuando el sistema OBD-II detecta algún problema a través de los transductores, se enciende una luz de advertencia en el tablero alertando al conductor de la falla existente. Además se guarda la información sobre las fallas detectadas en la memoria de la computadora del automóvil, facilitando al técnico automotriz encontrar los problemas para corregirlos posteriormente (Meseguer, 2013).

Los componentes del sistema OBD-II son: la ECU (*Engine Control Unit*) conocida como la computadora del automóvil, los transductores encargados de enviar los datos hacia la ECU, la luz indicadora de fallas (MIL, *Malfunction Indicator Light*) ubicado en el tablero, y el conector de diagnóstico (DLC, *Data Link Connector*) que sirve de interfaz entre la ECU y los dispositivos de diagnóstico automotriz.

A. Computadora del automóvil

La ECU es la computadora del automóvil y su función principal es obtener y manejar los datos provenientes de todos los transductores del motor del automóvil. Es un dispositivo que se encuentra generalmente debajo del tablero en la parte del conductor (Cervantes y Osborn, 2010).

B. Transductores del automóvil

Los transductores son dispositivos encargados de monitorear de forma continua el funcionamiento y operación del motor del automóvil [8]. Entre las medidas que pueden obtener los transductores del motor están: revoluciones por minuto del motor, temperatura del líquido refrigerante del motor, presión absoluta del colector de admisión, presión barométrica, temperatura de aire de admisión, posición del acelerador, velocidad del automóvil, entre otras.

En la Tabla 1 se presentan algunos transductores del motor del automóvil que permiten obtener las medidas antes mencionadas y que hacen que la ECU pueda determinar la cantidad de combustible, el punto de ignición y otros

parámetros necesarios que permiten evaluar las condiciones del automóvil (Zabler, 2002).

C. Luz indicadora de fallas

La luz MIL es utilizada por el sistema OBD-II y se enciende cuando los transductores del motor detectan un problema en el automóvil; su propósito es alertar al conductor de la necesidad de realizar un mantenimiento del mismo.

Tabla 1. Transductores del Motor del Automóvil

Sensor	Medida
Sensor CKP (<i>Crankshaft Position</i>)	Revoluciones por minuto
Sensor ECT (<i>Engine Coolant Temperature</i>)	Temperatura del refrigerante del motor
Sensor MAP (<i>Mainfold Absolute Pressure</i>)	Presión absoluta del colector de admisión
Sensor de presión barométrica	Presión barométrica
Sensor IAT (<i>Intake Air Temperature</i>)	Temperatura del aire de admisión
Sensor TPS (<i>Throttle Position Sensor</i>)	Posición del acelerador
Sensor VSS (<i>Vehicle Speed Sensor</i>)	Velocidad del automóvil

D. Conector DLC

El conector DLC es una interfaz con forma trapezoidal de 16 pines basado en el estándar SAE J1962 (McCord, 2011), que se ubica bajo el tablero, generalmente en el lado del conductor. El conector DLC sirve como interfaz de acceso y recuperación de datos desde la ECU hacia un equipo de diagnóstico (escáner automotriz) (McCord, 2011).

En la Tabla 2 se muestra la descripción de los 16 pines del conector DLC.

Tabla 2. Descripción de Pines del Conector DLC (Meseguer, 2013)

PIN	Características
1	Uso del fabricante
2	Bus (+) J1850 VPM y PWM
3	Uso del Fabricante
4	Tierra (chasis)
5	Señal de Tierra
6	Bus de datos CAN alto (J-2284)
7	Línea K ISO 9141-2
8	Uso del fabricante
9	Uso del fabricante
10	Bus (-) J1850
11	Uso del fabricante
12	Uso del fabricante
13	Uso del fabricante
14	Bus de datos CAN BAJO (J-2284)
15	Línea L ISO 9141-2
16	Voltaje de batería

E. Modos de medición

El sistema OBD-II utiliza 9 modos de medición, donde cada uno de los modos permite el acceso a los datos de la ECU del automóvil. Para solicitar datos de un automóvil es necesaria la utilización de códigos PID (*Parameter Identification*). Cada código PID está relacionado con una medida específica de los modos 1 y 2 del sistema OBD-II. Por ejemplo, si se desea solicitar el dato en tiempo real de la velocidad del automóvil, se debe ingresar al modo 1 y utilizar el PID 0D.

En la Tabla 3 se detallan las características de cada modo de medición soportados por el sistema OBD-II.

F. Comunicación de la ECU con dispositivos externos

En el sistema OBD-II, los protocolos de comunicación permiten establecer la comunicación e intercambio de mensaje de forma bi-direccional, entre una herramienta de diagnóstico (escáner automotriz) y la ECU del automóvil (García, 2013).

Los protocolos de comunicación soportados por el sistema OBD-II incluyen el SAE J1850 PWM (Modulación por ancho de pulso a 41.6 Kbps), el SAE J1850 VPW (Ancho de pulso variable a 10.4 - 41.6 Kbps), el ISO 9141-2 (Comunicación serial asincrónica a 10.4 Kbaud), el ISO 14230 KWP (Comunicación serial asincrónica hasta 10.4 Kbaud), y el ISO 15765 CAN (250 - 500 Kbps).

Tabla 3. Modos de Medición Soportados por el Sistema OBD-II (Meseguer, 2013)

Modo	Características
Modo 01	Obtención de datos actualizados.- Permite el acceso en tiempo real a los valores de salidas y entradas de la ECU.
Modo 02	Acceso a cuadro de datos congelados.- La ECU toma una muestra de los valores relacionaos con las emisiones en el momento exacto de ocurrir un fallo.
Modo 03	Obtención de los códigos de falla.- Permite extraer de la memoria de la ECU todos los códigos de fallo (DTC, <i>Data Trouble Code</i>) almacenados.
Modo 04	Borrado de códigos de falla y valores almacenados.- Se pueden borrar todos los códigos almacenados en la ECU, incluyendo los DTCs y el cuadro de datos grabado.
Modo 05	Resultado de las pruebas de los transductores de oxígeno.- Permite el acceso a los resultados de las pruebas realizadas a los transductores de oxígeno.
Modo 06	Resultados de las pruebas de otros transductores.
Modo 07	Resultado del diagnóstico en componentes que no están sujetos a vigilancia constante.
Modo 08	Muestra de códigos de falla pendientes.- Permite leer de la memoria de la ECU todos los DTCs pendientes.
Modo 09	Control de funcionamiento de componentes.-Se pueden realizar pruebas en los actuadores.
Modo 09	Información del automóvil.- Permite solicitar el número de identificación del automóvil (VIN, <i>Vehicle Identification Number</i>).

4. PROPUESTA DE ARQUITECTURA DE LA SOLUCIÓN TELEMÁTICA

La solución propuesta está conformada por un subsistema de transmisión y uno de recepción; estos se pueden apreciar en la arquitectura mostrada en la Figura 1.

A. Subsistema de transmisión

El subsistema de transmisión permite obtener los datos de los transductores del motor de un automóvil, genera la trama UDP con dichos datos y utilizando la red GPRS de una operadora móvil los envía hacia el servidor remoto. Este

subsistema, como se puede apreciar en la Figura 2, está conformado por tres bloques:

- Bloque de adquisición de datos.
- Bloque de procesamiento (generación de trama UDP) y transmisión de información.
- Bloque de alimentación.

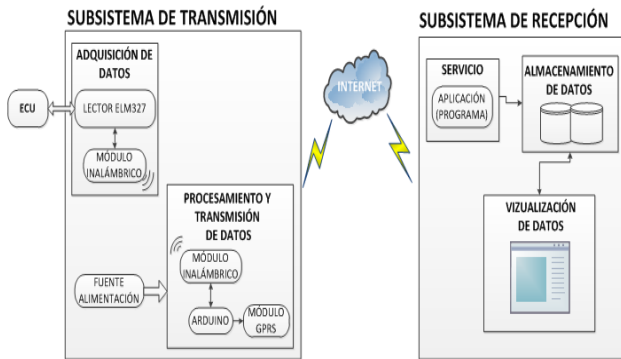


Figura 1. Arquitectura en alto nivel de la solución propuesta

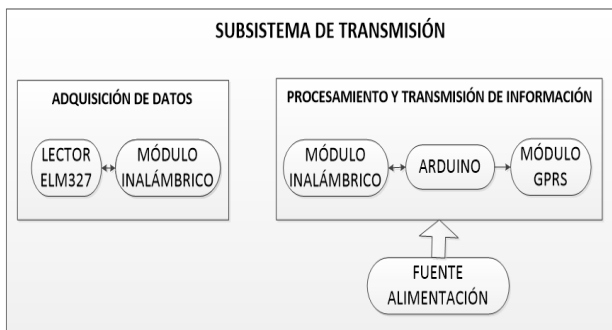


Figura 2. Bloques del subsistema de transmisión

1) *Bloque de adquisición de datos.*- Se basa en la utilización del lector de códigos OBD (ELM327), el cual está encargado de: conectarse a la ECU del automóvil a través del conector DLC, y solicitar los datos que producen los transductores del motor de un automóvil.

Esto se consigue debido a que soporta los protocolos de comunicación del sistema OBD-II.

Además, este bloque se conecta con el de generación de trama UDP y transmisión de información, a través de los módulos XBee serie 1, los cuales proporcionan una interfaz de conexión inalámbrica de corto alcance a bajo costo².

2) *Bloque de generación de trama UDP y transmisión de información.*- Se basa en la utilización del dispositivo Arduino Mega 2560 y el módulo SIM900 GSM/GPRS Shield mostrados en la Figura 3. Sus funciones son: inicializar el lector ELM327, seleccionar el protocolo de comunicación adecuado a través de comandos AT, solicitar los datos a la ECU y crear el mensaje UDP (*User Datagram Protocol*) a transmitir, cuyo formato se muestra en la Figura 4.

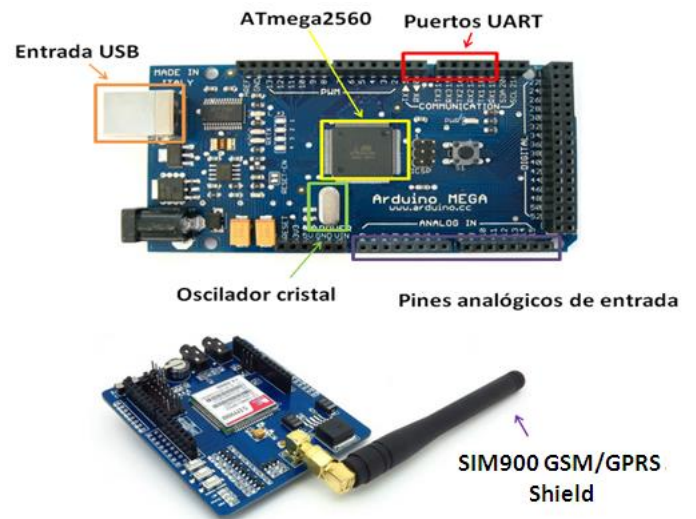


Figura 3. Dispositivo Arduino Mega 2560 y SIM900 GSM/GPRS Shield

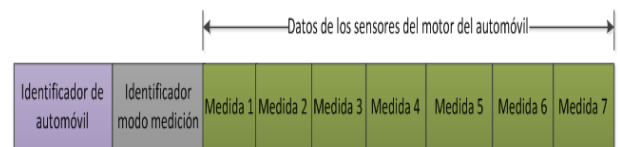


Figura 4. Formato del mensaje UDP

Cada campo del mensaje UDP tiene su respectiva función: el identificador de automóvil proporciona un número específico que se le asigna a cada automóvil de prueba, el identificador del modo indica en qué modo se trabaja y el resto son los valores medidos por los diferentes transductores del motor de un automóvil.

El bloque de generación de trama UDP y transmisión de información usa el SIM900 GSM/GPRS Shield, el cual es totalmente compatible con el dispositivo Arduino Mega 2560 y permite el envío del mensaje UDP hacia el servidor remoto³.

3) *Bloque de alimentación.*- Se encarga de ajustar los niveles de voltaje necesarios para energizar todos los dispositivos presentes en el bloque de generación de trama UDP y transmisión de información como: Arduino Mega 2560, al SIM900 GSM/GPRS Shield y al módulo XBee serie 1.

Para esto se utilizó un cargador USB (*Universal Serial Bus*) para automóviles, el cual se conecta directamente al dispositivo Arduino Mega 2560 a través de un cable USB, y proporciona el voltaje de 5V a 1A que es suficiente para alimentar a todos los elementos mencionados y tiene bajo costo.

Finalmente, en la Figura 5 se muestra el diagrama completo del subsistema de transmisión implementado.

2 <https://www.sparkfun.com/products/11215>

3 www.simcom.eu/index.php?m=termekek&prime=1&sub=40&id=0000000147

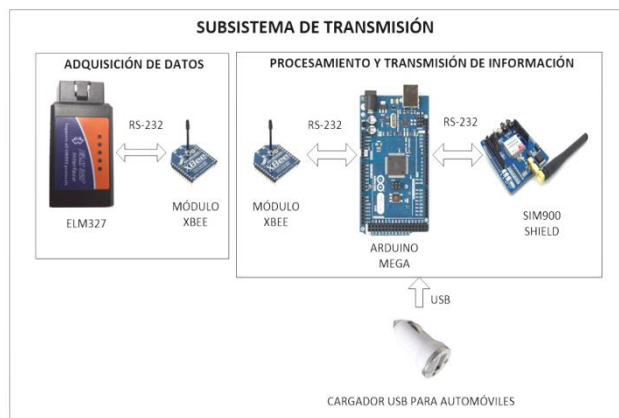


Figura 5. Subsistema de transmisión implementado

B. Subsistema de recepción

El subsistema de recepción permite recibir la información, almacenarla y comprobar, a través de una interfaz web, que la información llega correctamente. Este subsistema está conformado por los siguientes bloques:

- Bloque de servicio.
- Bloque de almacenamiento de información.
- Bloque de visualización de información.

1) *Bloque de servicio.*- Está constituido por un programa que se está ejecutando en segundo plano. Su principal función consiste en la recepción del mensaje UDP a través de un puerto asociado.

El programa fue desarrollado en lenguaje Java y se compone de tres etapas que son: apertura del puerto y lectura del mensaje, decodificación del mensaje UDP, conexión a la base de datos y almacenamiento de datos.

2) *Bloque de almacenamiento de información.*- Provee el medio para organizar y guardar toda la información de los valores medidos por los transductores del motor de un automóvil. Se utilizó el SGBD MySQL (*My Structured Query Language*).

Su función principal es almacenar registros en el transcurso del tiempo; esto permite realizar análisis estadísticos de los datos, a través de la creación de reportes. En la solución propuesta se usa una base de datos relacional orientada a comprobar la funcionalidad de la solución propuesta, cuyo diseño se presenta en la Figura 6.

3) *Bloque de visualización de información.*- Proporciona una interfaz web desarrollada con el lenguaje de programación PHP (*Hipertext Preprocessor*), que le permite al usuario verificar que los valores medidos por los transductores del motor de un automóvil llega y se almacenan correctamente.

El uso de la interfaz web garantiza el acceso a la información siempre que se cuente con una conexión a Internet y con un explorador web. La interfaz web se compone de 4 ventanas, cuyo modo de navegación se puede apreciar en el diagrama de bloques de la Figura 7.



Figura 6. Diagrama relacional de base de datos para el bloque de almacenamiento

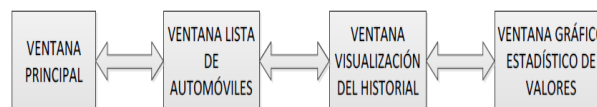


Figura 7. Distribución de las ventanas de la interfaz web básica

La ventana *visualización del historial*, es la interfaz más representativa, dado que presenta las características del automóvil y del dispositivo Arduino Mega 2560 utilizado, los valores medidos por los transductores del motor de un automóvil y que fueron almacenados anteriormente en la base de datos; y un filtro que permite escoger el parámetro deseado para visualizar y graficar dependiendo de la fecha inicial y final que se ingrese.

La ventana *gráfico estadístico de valores* presenta la curva histórica de los valores medidos por los transductores del motor de un automóvil, de acuerdo a la fecha inicial y fecha final que se ingresa en la ventana anterior.

C. Librería obd.h

La librería obd.h es de código abierto, creada por Stanley Huang⁴, y que tiene funciones que permiten el acceso a valores que son medidos por los transductores del motor de un automóvil provenientes de la ECU. Entre las principales características que presenta la librería obd.h están las siguientes:

⁴ Código fuente de la librería obd.h
<https://github.com/stanleyhuangyc/ArduinoOBD/blob/master/libraries/OBD/OBD.h>

- Permite configurar el dispositivo Arduino Mega 2560 para que trabaje a la misma velocidad de señal requerida por el lector ELM327 y soporte el modo 1 del sistema OBD-II, para poder solicitar los valores medidos por los transductores del motor de un automóvil en tiempo real.
- Incluye la librería `wire.h` que le permite al dispositivo Arduino Mega 2560 comunicarse con el lector ELM327, a través del uso de sus puertos UART (*Universal Asynchronous Receiver-Transmitter*).

Para garantizar el funcionamiento de la solución propuesta, es necesario configurar los siguientes parámetros:

- Cuando se utiliza el dispositivo Arduino Mega 2560, la librería `obd.h` configura el puerto UART 1 (puerto serial 1) para la transmisión y recepción de datos.
- La librería `obd.h` permite configurar la velocidad de señal a 38400 baudios, adaptándose sin ningún problema con la velocidad de señal del lector ELM327.
- La librería `obd.h` define un conjunto de constantes que representan de forma alfanumérica los diferentes PIDs del modo 1 del sistema OBD-II. Estos son indispensables para solicitar los valores medidos desde los transductores del motor del automóvil, por ejemplo: RPM del motor, velocidad del automóvil, posición del acelerador, presión barométrica, presión absoluta del colector de admisión, temperatura del refrigerante del motor y temperatura de admisión. La Figura 8 muestra ejemplos de la definición alfanumérica para algunos PIDs del modo 1 que puede soportar la librería `obd.h`.

```
#define PID_ENGINE_LOAD 0x04
#define PID_COOLANT_TEMP 0x05
#define PID_FUEL_PRESSURE 0x0A
#define PID_INTAKE_MAP 0x0B
#define PID_RPM 0x0C
#define PID_SPEED 0x0D
```

Figura 8. Ejemplos de representación alfanumérica de los PIDs soportados por la librería `obd.h`

La librería `obd.h` contiene varias funciones, de las cuales se han utilizado 3 de ellas y se describen a continuación:

- `begin()` se encarga de iniciar la comunicación entre el dispositivo Arduino Mega 2560 y el lector ELM327 a la velocidad de señal de 38400 baudios.
- `init()` permite al dispositivo Arduino Mega 2560 seleccionar el protocolo de comunicación del sistema OBD-II de forma automática.

- `read()` permite al dispositivo Arduino Mega 2560 leer los datos provenientes de la ECU del automóvil, dependiendo de los PIDs solicitados.

5. VALIDACIÓN Y RESULTADOS

La solución propuesta implementada puede ser utilizada en automóviles que cuentan con el conector DLC. Las pruebas se realizaron en dos automóviles de diferente fabricante, donde se logró verificar que no siempre se pueden obtener todas las medidas que producen los transductores del motor. Esto se debe a que antes de que el sistema OBD-II se estandarice, el fabricante utilizaba su propio criterio para la transmisión de datos, y por esta razón se producen incompatibilidades para la obtención de las medidas. La cantidad de medidas dependerá del grado de aplicación del estándar en los protocolos de los fabricantes.

En la Tabla 4 se presentan las características informativas de los automóviles usados para las pruebas de funcionamiento.

Tabla 4. Características de los Automóviles de Prueba

CARACTERÍSTICA	AUTOMÓVIL 1	AUTOMÓVIL 2
Marca	Ford Escape XLS	Hyundai
Año	2012	2009

Previo a las pruebas de funcionamiento, se realizó la conexión del lector ELM327 con el conector DLC, el cual generalmente se encuentra ubicado bajo el tablero en el lado del conductor, y cerca de los pedales.

El cargador USB de celulares del automóvil fungió como fuente de alimentación para el módulo XBee, el dispositivo Arduino Mega 2560, la pantalla LCD (*Lyquid Crystal Display*) y el SIM900 GSM/GPRS Shield.

La ubicación del dispositivo de recolección y transmisión de información va en la parte superior del tablero en los dos automóviles escogidos y su conexión directa con el cargador USB de celulares a través de un cable para su alimentación de voltaje.

A. Pruebas de la adquisición de datos

Las pruebas de adquisición de datos fueron enfocadas en comprobar el correcto funcionamiento del lector ELM327 con el automóvil. Se utilizó el Hyperterminal de la computadora para enviar de forma manual los comandos AT necesarios para solicitar los valores de un conjunto de PIDs del modo 1 del sistema OBD-II.

En la Figura 9 se muestra el ejemplo del envío del comando AT 01 05 por medio del Hyperterminal hacia el lector ELM327, donde el primer valor hace referencia al modo 1 y el otro valor representa al PID que solicita la temperatura del refrigerante del motor.

La respuesta en hexadecimal con el valor del PID que se recibe desde la ECU fue 41 05 7B, con la cual se genera la

trama UDP posteriormente por el dispositivo Arduino Mega 2560.

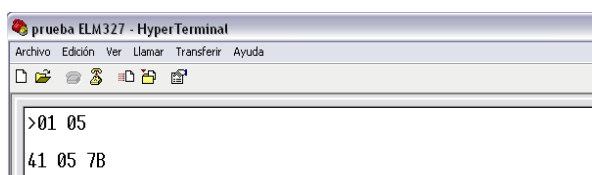


Figura 9 Ejemplo para solicitar la temperatura del refrigerante del motor de un automóvil

B. Pruebas en el Arduino Mega 2560 con la librería obd.h

Estas pruebas fueron enfocadas en verificar el funcionamiento del dispositivo Arduino Mega 2560, conjuntamente con la librería obd.h, para recibir las respuestas provenientes desde la ECU del automóvil y generar las tramas UDP. La respuesta obtenida por el Arduino se muestra a través de una pantalla LCD.

En la Figura 10 se muestra como ejemplo el resultado de la prueba realizada para la obtención de las revoluciones por minuto del motor. Se observa que la medida analógica mostrada en el tablero del automóvil es similar a la medida digital mostrada por la pantalla LCD de la solución propuesta.



Figura 10 Comparación del valor analógico del automóvil con el valor digital de la solución propuesta

Las pruebas realizadas dieron un número diferente de PIDs soportados en cada automóvil. La mayoría funcionó para el automóvil Ford Escape XLS, no siendo igual para el automóvil Hyundai Accent, debido a que el protocolo de comunicación que utiliza el fabricante Ford en sus automóviles, se adapta de mejor manera a los PIDs de la librería obd.h.

C. Pruebas de transmisión de información

Las pruebas de transmisión de información se enfocaron en verificar que el envío de información, desde el dispositivo Arduino Mega 2560 (en conjunto con el módulo SIM900 GSM/GPRS Shield), y la recepción en el servidor remoto se realizaron correctamente. El envío de información se hizo a

través de Internet, utilizando la red GPRS de la operadora Movistar.

La preparación previa a la realización de pruebas incluyó:

- La implementación del esquema de base de datos planteado previamente. El tener el servidor de la misma activo. Además, se ingresó directamente la información de identificación de los automóviles; importante ya que al momento de registrar la información de las medidas se verificará la integridad referencial.
- La implementación y despliegue del servidor programado en Java. Se usó el puerto UDP 9811. El servicio era accesible desde Internet, por lo que se hizo necesario el usar una IP (*Internet Protocol*) pública.
- El levantamiento de un servidor de páginas web, en este caso Apache, para poder acceder a la interfaz web.
- Se implementó una interfaz de terminal a través del software X-CTU para verificar los comandos AT que se iban ejecutando⁵.

Es importante mencionar que la ventana del terminal del software X-CTU permite al usuario visualizar la lista de comandos AT que ejecuta el dispositivo Arduino Mega 2560 para el envío del mensaje UDP creado, hacia el servidor remoto.

En el programa servidor se visualizaron los mensajes UDP que llegaron provenientes del subsistema de transmisión. En la Tabla 5 se muestra la ventana de la interfaz web que contiene todos los valores medidos por los transductores del motor del automóvil Ford Escape XLS.

La interfaz web colorea de rojo a los valores que sobrepasan un valor máximo que ha sido definido en el código de la aplicación.

Cabe mencionar que ese valor máximo simplemente sirve de prueba para comprobar el correcto funcionamiento de la interfaz web. Para obtener un control preventivo real con la solución propuesta, se debería obtener soporte de personal calificado, quién sería el encargado de colocar los rangos de valores reales que necesita el automóvil para que el funcionamiento del motor sea el correcto.

En la Tabla 6 se puede observar un ejemplo en el cual se considera que el valor máximo de la velocidad es 50 Km/h, y que aquellos valores que sobrepasen ese límite se pongan de color rojo.

Esto puede servir como un registro de que el conductor cometió una infracción de tránsito en caso de haber estado circulando en el perímetro urbano.

⁵ <http://www.digi.com/support/productdetail?pid=3352>

Tabla 5 Registro de valores del automóvil Ford Escape XLS

Id registro	Modo medición	RPM (Rpm)	Velocidad (Km/h)	Pos acelerador (%)	Presión barométrica (kPa)	Presión abs_colect kPa	Temp refrigerant (°C)	Temp admisión (°C)	Temp ambiente (°C)
163	1	1637	31	18	76	38	97	27	21
162	1	1148	5	19	76	50	96	28	21
161	1	651	0	14	76	31	95	28	21
160	1	900	6	15	76	27	95	28	21
159	1	1125	43	15	76	21	96	27	21

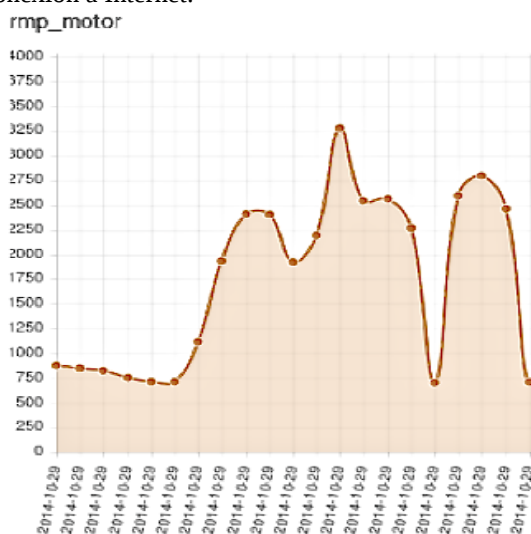
En la Figura 11 se muestra la gráfica estadística obtenida con las medidas de las revoluciones por minuto, sin embargo la interfaz web está programada para que se pueda mostrar la gráfica estadística de cualquier otra medida.

Tabla 6 Ejemplo de valores no permitidos de la velocidad del automóvil

Id registro	Modo medición	RPM (Rpm)	Velocidad (Km/h)	Pos acelerador (%)	Presión barométrica (kPa)
163	1	1637	31	18	76
162	1	1148	5	19	76
161	1	651	0	14	76
160	1	900	6	15	76
159	1	1125	43	15	76
158	1	1405	51	17	76
157	1	1649	61	18	76
156	1	1370	50	18	76

Valores no permitidos

Finalmente, se debe mencionar que el usuario tiene la facilidad de ir monitoreando los valores medidos por los transductores del motor de un automóvil no solo por medio de una computadora, sino también puede utilizar otros dispositivos como celulares o tabletas siempre que cuenten con conexión a Internet.

**Figura 11** Gráfica estadística de las revoluciones por minuto

D. Cálculo de tráfico

La solución propuesta trabaja con la arquitectura TCP/IP sobre Internet, donde el protocolo de capa transporte que se utilizó fue UDP. A continuación se procede a realizar el cálculo del tráfico producido durante la transmisión de la información; los resultados obtenidos serán de utilidad para estimar costos de comunicación al utilizar la red de una operadora celular.

La cantidad de bytes utilizados para los mensajes UDP a enviarse es de 35 bytes, que se obtienen del identificador del automóvil (1 byte), del modo de medición (1 byte), de las 7 medidas obtenidas de los transductores del motor de un automóvil (22 bytes) y del separador de cada dato (11 bytes).

Capa de transporte:

Bytes de información: 35 bytes.

Bytes de cabecera para mensajes UDP: 8 bytes.

Total bytes del mensaje UDP: 43 bytes.

Capa de Internet:

Bytes de información: 43 bytes.

Bytes de cabecera para datagrama IP: 24 bytes.

Total bytes para datagrama IP: 67 bytes.

La capa de acceso a red de arquitectura GPRS se compone de dos subcapas: LLC (*Logical Link Control*) y RLC/MAC (*Radio Link Control / Medium Access Control*), donde cada uno tiene su propio formato de trama.

Subcapa LLC:

Bytes de información: 67 bytes.

Bytes de cabecera LLC: 37 bytes.

Bytes de cola LLC: 3 bytes.

Total bytes para LLC: 107 bytes.

Para determinar los bytes de la trama de la subcapa RLC/MAC, se debe considerar la codificación que se realiza en la capa física de la arquitectura GPRS. Para el cálculo se consideran los dos escenarios extremos de codificaciones que son CS-1 y CS-4.

Subcapa RLC/MAC para codificación CS-1:

Bytes de información: 107 bytes.

Número de segmentos: 107 bytes/23bytes

Número de segmentos: 5 segmentos
Bytes total información: 5*57 bytes
Bytes de información: 285 bytes

Subcapa RLC/MAC para codificación CS-4:
Bytes de información: 107 bytes.
Número de segmentos: 107 bytes/53bytes
Número de segmentos: 2 segmentos
Bytes total información: 2*57 bytes
Bytes de información: 114 bytes

Considerando un caso en el cual se envíe la información cada minuto, se calcula el total de información que se produce en un día y a su vez el que se producirá en un mes, teniendo en cuenta que el funcionamiento de la solución propuesta es continuo.

Codificación CS-1:
Bytes información en minutos: 285 bytes.
Bytes información en días: $285 \times 60 \times 24 = 410400$ bytes.
Bytes información mensual: $410400 \times 30 = 12312000$ bytes.
Bytes información mensual: 11.74 MBytes.

Codificación CS-4:
Bytes información en minutos: 114 bytes.
Bytes información en días: $114 \times 60 \times 24 = 164160$ bytes.
Bytes información mensual: $164160 \times 30 = 4924800$ bytes.
Bytes información mensual: 4.70 Mbytes.

6. TRABAJO FUTURO

El trabajo propuesto actualmente sólo realiza mediciones en tiempo real o en modo 1, dado que la librería utilizada OBD.h sólo soporta este modo de obtención de información. Un trabajo futuro podría ser la actualización de la librería para que soporte otros modos en la lectura de parámetros del vehículo, como el modo de fallos.

Adicionalmente, un proyecto de mejora sobre el trabajo actual implica trabajar en la independencia de energía del dispositivo de obtención de datos, dado que al momento ésta depende de la alimentación eléctrica del vehículo. El trabajo futuro implicaría el investigar en una batería que de una independencia temporal de energía del vehículo para proporcionar seguridad e independencia del lugar de instalación del dispositivo dentro del vehículo.

La presente implementación envía la información de un vehículo dependiendo de la red GPRS, en el caso de no detectar señal, el dispositivo no estaría en capacidad de enviar la información al servidor remoto. Un trabajo de mejora se podría enfocar en la grabación temporal de información en una memoria incluida en el Arduino, en caso de que la red GPRS no se encuentre disponible, y una vez que se detecte que la red puede ser utilizada, se procedería a realizar el envío de datos al servidor remoto.

7. CONCLUSIONES

Se pudo adaptar el dispositivo ELM327 para que envíe y reciba información a través de una interfaz ZigBee.

Se comprobó que un Arduino Mega 2560 resulta ser bastante versátil para aplicaciones de envío de datos remotos. Dicha versatilidad se alcanza por la cantidad de puertos disponibles que tiene, y que pueden ser conectados a diferentes Shields de comunicaciones.

La cantidad de PIDS soportados por un vehículo depende del protocolo de comunicación que este utilice, y del grado de adopción del estándar OBD-II que tenga.

La cantidad de datos que se consumiría mensualmente (en procesos de lectura de medidas realizados cada minuto) es bajo, lo cual implica que es viable el realizar proceso de seguimiento de este estilo, sin que esto incurra en el pago de grandes tarifas por consumo de datos. Incluso se podría mejorar el consumo de datos, al pensar en la formación del mensaje a enviar con bits en lugar de con caracteres.

Los elementos utilizados y propuestos en la arquitectura permitieron leer, recolectar, enviar remotamente, y almacenar los datos que se requerían del sistema OBD-II.

REFERENCIAS

- Baghli, L., Moussaoui, A., Benmansour, K., Delprat, S., & Djemai, M. (2012, June). Optimal hybrid vehicle, embedded data acquisition and tracking. *Environment Friendly Energies and Applications (EFEA)*, 2012 2nd International Symposium on (pp. 68-73). IEEE.
- Baghli, L., Benmansour, K., & Djemai, M. (2014, November) Development of a data acquisition and tracking system for vehicles. *Environmental Friendly Energies and Applications (EFEA)*, 2014 3rd International Symposium on. IEEE.
- Cervantes Alonso, Espinosa Solís, S. O. (2010). Escaner Automotriz de Pantalla Táctil (Doctoral dissertation).
- Gamec, J. (2012). Wireless real-time vehicle monitoring based on android mobile device. *Acta Electrotechnica et Informatica*, 12(4), 7.
- Lee, M. G., Park, Y. K., & Jung, K. K. (2014). Intelligent Prediction Model of Vehicle Fuel Consumption. *International Information Institute (Tokyo)*. Information, 17(5), 1825.
- McCord, K. (2011). Automotive Diagnostic Systems. CarTech Inc.
- Meseguer Anastasio, J. E. (2013). Caracterización de los estilos de conducción mediante smartphones, dispositivos obd-ii y redes neuronales.
- Montero, G., & Sebastián, M. (2013). Elaboración de un Manual Técnico para el Análisis del Diagnóstico Electrónico de Vehículos con Sistema OBD-II para el Taller Ambamazda SA Durante 1 año 2012.
- Niazi, M., Nayyar, A., Raza, A., Awan, A. U., Ali, M. H., Rashid, N., & Iqbal, J. (2013, December). Development of an On-Board Diagnostic (OBD) kit for troubleshooting of compliant vehicles. In *Emerging Technologies (ICET)*, 2013 IEEE 9th International Conference on (pp. 1-4). IEEE.
- Zabler, E. (2002). Los sensores en el automóvil. Reverte.